

DataLabs API Integration Guide Compliance 4.120

February 2023

Copyright, 2023 Behavox - This document and its content are protected by Canadian copyright law. Except as otherwise provided for under Canadian copyright law, this document and its content may not be copied, published, distributed, downloaded or otherwise stored in a retrieval system, transmitted or converted, in any form or by any means, electronic or otherwise, without the prior written permission of the copyright owner.

Contents

- DataLabs API overview..... 1**
- DataLabs API artifacts..... 3**
- DataLabs API onboarding..... 6**
- DataLabs API rate limiting and thresholds..... 7**
- DataLabs API authorization and authentication..... 7**
 - Access token policies and access tokens..... 8
 - Creating an access token policy.....8
 - Deleting an access token policy.....9
 - Reviewing active access tokens.....9
 - Revoking an access token..... 10
 - Creating an access token..... 10
 - Revoking your own access token.....11
- DataLabs API best practices.....11**
 - Paginating Behavox Compliance API returns..... 11
 - Linking data between DataLabs API endpoints..... 14
- DataLabs API troubleshooting.....19**

DataLabs API overview

The DataLabs API enables developers to extract enriched data from Behavox Compliance for use in third-party applications.

The Behavox data ingestion process presents a unique opportunity for clients to use a single trusted source of content and metadata. This opportunity can address the following business challenges:

- Efficiently aggregating unstructured communications data that is stored in different systems across the enterprise
- Providing AI-driven insights, structured communications data, and metadata to third-party applications like BI tools and CRM
- Integrating additional surveillance platforms for regulatory compliance

What is enriched data?

Behavox Compliance enriched data includes:

- AI insights, including alerts
- Metadata for communications
- Structured artifacts, including ticker symbols, signatures, and deal indicators
- Profiles, including alert reviewers and persons

API endpoints: datalabs definition

The DataLabs API supports a range of endpoints to support your Behavox Compliance integration.


 **Note:** The API supports only the **GET** method for all endpoints.

Group	Endpoint	Description
activity-logs		
	/api/1/activity-logs	Returns a list of activity logs as seen on Behavox Compliance. Activity log actions are paginated into fixed number of items defined by the limit parameter.
alerts		

Group	Endpoint	Description
	/api/1/alert	Returns alert details, participants, and justifications, for a given alert ID.
	/api/1/alerts	Returns alert details, participants, and justifications. Alerts are paginated into a fixed number of items defined by the limit parameter.
	/api/1/escalation-levels	Returns the list of escalation levels configured in the system.
	/api/1/resolution	Returns the list of resolutions configured in the system.
	/api/1/risk-policies	Returns the list of risk policies configured in the system.
Communications-V3		
	/api/3/artifacts	Returns ML generated insights from the communication data on the Behavox platform.
	/api/3/communications	Returns communication data for given IDs.
	/api/3/communications/find	Returns communication data including participants involved, alerts flagged, and data mining artifacts found, such as ticker symbols, organizations, and signatures. In the response, communications are paginated into fixed number of items defined by the limit parameter.
	/api/3/data-types	Returns the available data types.
	/api/3/features	Returns derived fields from the communication data on the Behavox platform.
persons		
	/api/1/person	Returns person data for given person ids, including contact information and profile attributes as seen on Behavox Compliance.
	/api/1/person-groups	Returns a list of person groups configured in the system.
	/api/1/persons	Returns person data including contact information and profile attributes as seen on Behavox Compliance. In the response, persons are paginated into fixed number of items defined by the limit parameter.

DataLabs API artifacts

The Behavox Compliance data enrichment process identifies artifacts and structures their content so that it can be used in third-party applications. Artifacts are identified within email and chat data types.

 **Note:** Artifacts are not supported if you are using the **communications** definition.

Artifacts are named entities that are located and classified within the unstructured text of emails and chats. The following entities are recognized as artifacts:

- Ticker
- Organization
- Deal indication
- Signature
- Question

Ticker

Ticker symbols identified within an email or chat are structured with the following data schema:

Field	Data type	Nullable	Example	Description
ticker	string	false	APPL	Ticker symbol
instrument-Type	string	false	equity (eq), fixed in- come (fi), foreign ex- change (fx), commodity (com), index (ind)	Financial instru- ment type
position	string	true	Buy,sell,both, buy_past,sel- l_past	Trading position
maturity	string	true	10/02/2023, 02/2023, 5Y	Maturity date. The date can take various for- mats.
interestRate	string	true	9.00	Interest rate

Field	Data type	Nullable	Example	Description
notional	string	true	1000.00	Trade notional amount.
askPrice	string	true	11.000	Ask price - the price for which trader will sell a security.
bidPrice	string	true	11.000	Bid price - the price a trader will pay for a security.
sentiment	string	true	bullish, bearish, neutral, bullish_rating, bearish_rating, neutral_rating	Sentiment analysis on the surrounding context of ticker.
buyAmount	string	true	2400.00	Buy amount
sellAmount	string	true	2400.00	100
exchange-Symbol	string	true		Sell amount

Organization

Organizations identified within an email or chat are structured with the following data schema:

Field	Data type	Example	Description
organization	string	CNBC	

Deal indication

Deal indicators identified within an email or chat. Deal indicators are unique as they combine a named entity and sentiment analysis. Examples:

- **tradeConfirmation** – Confirmation of a deal that is closed (deal can be confirmed either by client or salesperson).
- **color** – Market color, salesperson shows client what he/she is doing in the market. Also client can share market information on some instruments with salesperson that he / she has good relationship with.
- **clientBite** – Client demonstrates an interest after salesperson pitched an instrument.
- **clientRejection** – Client rejects a pitch.
- **clientInterest** – Client comes to a salesperson with indication of interest in particular instrument.
- **clientComesWithOrder** – Client comes to a salesperson and leaves an order.
- **salesPitch** – Salesperson pitches an instrument to a client.
- **clientLeavesOrder** – Client leaves an order after salesperson pitch.

Deal indicators are structured with the following data schema:

Field	Data type	Example	Description
tickers	string array	[APPL, GOOG]	
type	string	tradeConfirmation, color, clientBite, clientRejection, clientInterest, clientComesWithOrder, salesPitch, clientLeavesOrder	Sentiment analysis on the surrounding context of ticker.

Signature

Signatures identified within an email or chat are structured with the following data schema:

Field	Data type	Example	Description
jobTitles	string		
emails	string		
phoneNumbers	string		

Field	Data type	Example	Description
personNames	string		
organizations	string		
language	string	en	Language of signature

Question

Questions within an email or chat are structured with the following data schema:

Field	Data type	Example	Description
question	string	Do you have chewing gum?	

DataLabs API onboarding

The DataLabs API integration requires very little preparation. This workflow details the Behavox Compliance data-enrichment process and the user preparation required to query enriched data.

Behavox data enrichment

Behavox completes the following steps during data enrichment:

1. Data ingestion – Content and HR profile data is ingested and stored in a database.
2. Data capture – Content, persons and logging data is identified and captured from the database.
3. Insight generation – Alerts, justifications, features, and artifacts are generated and stored in the database.
4. IP whitelisting – Customer provides IP addresses for whitelisting by Behavox to enable calls to the Behavox Compliance API.

Behavox Compliance API preparation workflow

You must complete the following steps to use API endpoints to query enriched data:

1. A Behavox Compliance administrator creates a user role with **API_user** data level access, including:
 - All content types
 - All alerts
 - All persons
 - All log events
2. A Behavox Compliance administrator assigns the **API_user** role to the profile of an API developer.
3. A Behavox Compliance administrator configures an access token policy to set token expiry periods for the **API_user** API developer access tokens.
4. The API developer creates an access token to authorize their API calls.
5. The API developer creates queries, using DataLabs API endpoints.

DataLabs API rate limiting and thresholds

The DataLabs API integrations are subject to the following rate limits and thresholds.

DataLabs API rate limits:

- 10 requests/second for an IP address
- 20 requests/second globally



Note: The DataLabs API will return an error code 439 when rate limits are exceeded.

DataLabs API authorization and authentication

The use of DataLabs API integrations is authorized and authenticated through access tokens that are passed to the endpoints in REST message headers. Access tokens are generated and managed for an API developer on Behavox Compliance.

You must include your access token in the **Authorization** header as a bearer token, as detailed in the following example.

Get all the risk policies

```
curl -X 'GET' \
  'http://dev-doc-1230-cluster.aws.internal:8080/dashboard/api/1/risk-policies' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer GCa8wGw0NA8LIqs6V4lPaW5u4fLUNyrGYQhbpPh1qvM'
```

Access token policies and access tokens

DataLabs API access token policies enable an administrator to configure an expiry period for access tokens. Access token policies are created in the Behavox Compliance administrator panel. An API developer can create an access token from their profile for use in the **Authorization** header as a bearer token.

Ensuring that access tokens expire is an important security measure to mitigate the risk of leaked access tokens.

Access token policies control the expiry period for a token generated by an API user, but an administrator can revoke an access token at any time.



Note: The default expiry for an access token is 90 days.

An administrator can:

- Create access token policies
- Delete access token policies
- Review active access tokens generated by API users
- Revoke an access token, and prohibit an API user

An API user can:

- Create an access token
- Revoke their own token

Creating an access token policy


Administrators can configure an expiry period for access tokens in Behavox Compliance.

About this task

At the **Administrator panel**:


Procedure

1. Open the policy editor. Click **Application > Access token policies**.
2. Click **New access token policy**.
3. Set a period for token expiry.
 - a. Enter an integer in the **Expiration time** dialog box.
 - b. Select a unit of time from **Day(s)**, **Week(s)**, **Year(s)**.

 **Note:** The system determines an expiry date for each token based on the selected period for the policy. Creation and expiration dates for a token are visible in the **Active access tokens** panel.

4. Commit your changes. Click **Save**.

What to do next

 **Note:** Only one access token policy may be active at a time. To change the policy, you must delete it and create a new one.

Deleting an access token policy

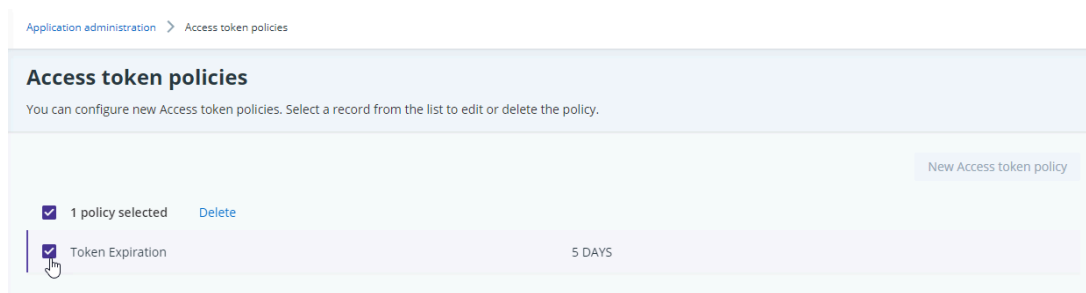
You can delete an existing access token policy.

About this task

At the **Administrator panel**:

Procedure

1. Open the policy editor. Click **Application > Access token policies**.
2. Select the policy. Select the check box for the policy record from the list.



3. Delete the record. Click **Delete**.

Reviewing active access tokens


Behavox Compliance administrators can review active DataLabs API tokens.

About this task

At the **Administrator panel**:

Procedure

Open the **Active access tokens** panel. Click **Application > Active access tokens**

 **Note:** Only active tokens are displayed. The system does not support a registry of revoked tokens.

Revoking an access token

Behavox Compliance administrators can revoke active DataLabs API tokens.

About this task

At the **Administrator panel**:

Procedure

1. Open the **Active access tokens** panel. Click **Application > Active access tokens**.
2. Revoke the access token. Select the check box for the token and click **Revoke**.

Creating an access token


An API user can create an access token to use in the DataLabs API. Use an access token in the **Authorization** header as a bearer token.

About this task

At your Behavox Compliance desktop:

Procedure

1. Open your user profile.
 - a. Click the icon/avatar on the upper-right corner of the banner.
 - b. Click the **Settings** link in the profile card.
2. Open the **Access Tokens** tab. Click the tab.
3. Configure a new access token. Enter a name in the **Token name** dialog box and click **Generate**.
4. Copy and save your token in a secure space.

- a. Click the Copy button  to place the token string on your clipboard.
- b. Paste the token in a secure place.

Revoking your own access token

You can revoke an access token that you have created for your own use.

About this task

At your Behavox Compliance desktop:

Procedure

1. Open your user profile.
 - a. Click the icon/avatar on the upper-right corner of the banner.
 - b. Click the **Settings** link in the profile card.
2. Open the **Access Tokens** tab. Click the tab.
3. Revoke the access policy. Select the check box for the token and click **Revoke**.

DataLabs API best practices

The DataLabs API best practices are collection of high-value developer activities that Behavox recommends to ensure your integrations are fast and secure.

Consider the following tips when creating your integration.

Scheduling automated queries

Ensure your API automation does not schedule processes during the Behavox data ingestion cycle. Conflicting processes may cause service degradation.

Paginating Behavox Compliance API returns

Paginate your Behavox Compliance API returns to ensure the best performance from your integration. Pagination breaks up a large set of returns into a series of smaller returns.

The DataLabs API implements a specific type of pagination called cursor pagination. Cursor-based pagination works by returning a pointer to a specific item in the dataset. On subsequent requests, the server returns results after the given pointer. This method addresses the drawbacks of using offset pagination, but does so by making certain trade offs:

- There is no concept of the total number of pages or results in the set.
- The client can't jump to a specific page.

Paged responses include cursor data. The last page in the result set contains a null value for the cursor, there are no more returns. The following figure details the first page of a return with cursor data.

```
{
  "alerts": [
    {
      "alertId": 0,
      "contentIds": [
        "lSnv3Q:E"
      ],
      "creationDate": "2015-07-20T15:49:04-07:00",
      "closingDate": "2015-07-20T15:49:04-07:00",
      "alertName": "Conflicts and Disputes",
      "status": "Created",
      "category": "Other",
      "severity": "HIGHEST",
      "ownerId": 0,
      "level": "L1",
      "resolution": "NOT_ISSUE",
      "creatorId": 0,
      "meParticipants": [
        0
      ],
      "justifications": [
        null
      ],
      "comments": [
        {
          "id": 24,
          "userId": 213,
          "date": "2015-07-20T15:49:04-07:00",
          "text": "string"
        }
      ]
    }
  ]
}
```

```

    }
  ],
  "pagination": {
    "cursor": "doc#kncPTg:P3"
  }
}

```

To request the next set of returns, send a request with the cursor data `doc#kncPTg:P3`, as detailed in the following figure.

```

https://virtserver.swaggerhub.com/Behavox/api/1/communications?start-date=2022-05-03T17%3A30%3A42.424Z&end-date=2022-05-03T17%3A30%3A42.424Z&data-type=email&artifact-type=organization&person-group=RandomSamplingGroup&cursor=doc%23kncPTg%3AP3&limit=100

```

The following figure details the final return in a set with a null value for the cursor.

```

{
  "alerts": [
    {
      "alertId": 0,
      "contentIds": [
        "lSnv3Q:E"
      ],
      "creationDate": "2015-07-20T15:49:04-07:00",
      "closingDate": "2015-07-20T15:49:04-07:00",
      "alertName": "Conflicts and Disputes",
      "status": "Created",
      "category": "Other",
      "severity": "HIGHEST",
      "ownerId": 0,
      "level": "L1",
      "resolution": "NOT_ISSUE",
      "creatorId": 0,
      "meParticipants": [
        0
      ],
      "justifications": [
        null
      ],
      "comments": [

```



```

    {
      "id": 24,
      "userId": 213,
      "date": "2015-07-20T15:49:04-07:00",
      "text": "string"
    }
  ]
},
{
  "pagination": {
    "cursor": "null"
    "total": 0
  }
}

```

 **Note:** All URLs are subject to encoding to restrict the character set to ASCII. For example, the pagination cursor `Sdg7rg:E` appears as `Sdg7rg%3AE` in an URL.

Linking data between DataLabs API endpoints

Internal IDs are provided in responses to link data between endpoints. This enables an API developer to reference data between alerts, persons, and content.

Alert IDs

Alert IDs appear in responses to a query for communications data if the content is associated with an alert.

```

[
  {
    "id": "Sdg7rg:E",
    "dataType": "email",
    "alertIds": [
      221
    ],
    "subject": null,
    "date": "2002-01-30T16:25:36Z",
    "participants": [

```


Once you have determined the alert ID, you can use it to query for the corresponding alert from the **/alert** endpoint.

In this example, we will use the alert ID from the communications data to pull the alert details using `GET /api/1/alert?alertId=221`

```
[
  {
    "alertId": 221,
    "contentIds": [
      "Sdg7rg:E"
    ],
    "creationDate": "2022-08-15T08:26:54Z",
    "closingDate": null,
    "alertName": "Random Sampling - All",
    "status": "CREATED",
    "category": "Information Security",
    "severity": "ZERO",
    "ownerId": null,
    "level": "L1",
    "resolution": null,
    "creatorId": null,
    "meParticipants": [
      37
    ],
    "justifications": [],
    "comments": []
  }
]
```

Communications/content ID

Content IDs appear in the response to an alert query.

 **Note:** Communications IDs and content IDs are the same entities, references to either are equivalent. Content IDs should be URL-encoded.

```
[
  {
```

```
"alertId": 221,
"contentIds": [
  "Sdg7rg:E"
],
```


We can use the content ID to get the corresponding communication from the **/communication** endpoint.

In this example, we will use the content ID from the alert data to pull the content details using `GET /api/1/communication?id=Sdg7rg%3AE`.

```
[
  {
    "id": "Sdg7rg:E",
    "dataType": "email",
    "alertIds": [
      221
    ],
    "subject": null,
    "date": "2002-01-30T16:25:36Z",
    "participants": [
      {
        "personId": 37,
        "name": "Steven Kean",
        "phone": "tel:+5533288161",
        "email": "steven.kean@enron.com",
        "company": "Enron",
        "type": "internal",
        "monitored": true,
        "personGroups": [
          "RandomSamplingGroup2"
        ],
      },
    ],
```

Person/participant ID

Person IDs are returned in both communication responses and alert responses.

 **Note:** Person IDs and participant IDs are the same entities, references to either are equivalent.

Communication:

```
[
  {
    "id": "Sdg7rg:E",
    "dataType": "email",
    "alertIds": [
      221
    ],
    "subject": null,
    "date": "2002-01-30T16:25:36Z",
    "participants": [
      {
        "personId": 37,
        "name": "Steven Kean",
        "phone": "tel:+5533288161",
        "email": "steven.kean@enron.com",
        "company": "Enron",
        "type": "internal",
        "monitored": true,
        "personGroups": [
          "RandomSamplingGroup2"
        ],
      },
    ],
  },
]
```

Alert:

```
[
  {
    "alertId": 221,
    "contentIds": [
      "Sdg7rg:E"
    ],
    "creationDate": "2022-08-15T08:26:54Z",
    "closingDate": null,
    "alertName": "Random Sampling - All",
    "status": "CREATED",
    "category": "Information Security",
    "severity": "ZERO",
  },
]
```

```

"ownerId": null,

"level": "L1",

"resolution": null,

"creatorId": null,

"meParticipants": [

  37

],

```

We can use the Person ID to identify the ME using the **/person** endpoint.

In this example, we will use the Person ID from the alert data to pull the ME details using `GET /api/1/person?personId=37`.

```

[
  {
    "personId": 37,
    "firstName": "Steven",
    "lastName": "Kean",
    "type": "internal",
    "contacts": [
      {
        "type": "EMAIL",
        "value": "kean@rice.edu",
        "primary": null
      },
      {
        "type": "PHONE",
        "value": "tel:+5533288161",
        "primary": null
      },
      {
        "type": "CHAT",
        "value": "296843139493924",
        "primary": null
      },
    ],
  },
]

```

 **Note:** All URLs are subject to encoding to restrict the character set to ASCII. For example, the pagination cursor `Sdg7rg:E` appears as `Sdg7rg%3AE` in an URL.

DataLabs API troubleshooting

The API is supported with URL response codes.

HTTP response Code	HTTP response message
200	OK - the request succeeded.
400	Bad request - The server cannot or will not process the request due to something that is perceived to be a client error (e.g., malformed request syntax, invalid request message framing, or deceptive request routing).
401	Unauthorized - Access token is missing or invalid
403	Forbidden - The endpoint is not enabled. Unlike 401 Unauthorized , the client's identity is known to the server.
439	Rate limits exceeded.

www.behavox.com

BEHAVOX